Consultingwerk
software architecture and development

Say Bye-Bye to the IDE. Here Comes Claude Code.

Mike Fechner
Consultingwerk

The full stack modernization framework
SmartComponent Library
by Consultingwerk

Mike Fechner



- 36 years of experience in Progress (from Version 5 to OpenEdge 13)
- Active member of the OpenEdge community and speaker at international conferences
- Expert in software architecture, object-oriented design, and web technologies

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Consultingwerk
software architecture and development

Our Focus: Modernizing OpenEdge Applications



Modernization of Legacy OpenEdge Applications



Deep Technical Expertise



Global IT Partner with Local Presence



More than Consulting – We Deliver Tools & Solutions

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.



How We Got Here

From smart autocomplete to a coding agent that knows ABL

Our Journey with AI Developer Tools



Spring 2024

Windsurf

AI-enabled VS Code: smarter proposals, variables and loops typed for us

2025

The challenge

Customers and CIOs ask whether OpenEdge is still the right platform

Mid 2025

Reality check

Mainstream AI still struggles on stage

2025/26

Turning point

Strong models and agents, plus ABL knowledge from Progress

2026

AI first

Claude Code writes our code — no IDE needed

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Early 2025: Is OpenEdge Still the Right Platform?



"Everybody else develops faster with AI — and AI doesn't work with OpenEdge."



Who asked

- Developers and CIOs in modernization projects
- Fuelled by IT press: "mainstream stacks code with AI"



What was at stake

- The choice of platform for the next decade
- Modernization projects questioned before they start

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Mid-2025 Reality Check: WeAreDevelopers Conference

10,000+

developers in the keynote audience

30 min

keynote by GitHub CEO Thomas Dohmke

0

working Snake games prompted live — in a mainstream stack

GitHub = Copilot = Microsoft: mainstream big tech.
So much for “everybody is already replacing developers with AI” in mid-2025.

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

The Turning Point at the Turn of the Year 2025/26

Models and agents grew up

- LLMs grasp complex logic and design
- Coding agents capable of nearly any complexity
- Claude Opus, GPT and others: ABL out of the box

Progress closed the ABL gap

- OpenEdge AI Assistant (formerly MCP Connector for ABL)
- Language reference and Knowledge Base for any coding agent

The result

AI is finally productive for developers — in ABL and in mainstream stacks alike

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Why ABL in Times of AI?

NEW

Easy for Humans and Machines

- Verbose, close to natural language
- Built for business people to write business applications — 40 years ago
- The same property makes it very accessible to GenAI

Faster in ABL than in Angular

- Our experience in real projects
- Comparable tasks: the ABL side is done before the Angular side
- OpenEdge is not behind when it comes to AI

Reviewable at 10× Speed

- 10× the code still needs a human reviewer
- ABL reads easier than nested anonymous functions and arrow chains
- Readable code is the bottleneck now — not typing

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Confessions of a Tool Enthusiast



Championed it as the modern OpenEdge tool

OpenEdge Architect and Developer Studio: early adopter and evangelist

Early adopter for OpenEdge development

Spring 2024: AI features in the IDE

2026: no IDE needed anymore

Always the tool enthusiast — and in 2026 the best tool turned out to be no IDE at all

"I stopped being a Progress programmer ten months ago.
Now I'm just a herder of AI tools."

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Today



Many days I produce tons of source code
and never see an IDE once

Tools I see more often than VS Code



Jira

Although very often through Claude



GitKraken

Code reviews during code generation and before the PR

The agent edits, compiles, tests and commits. I specify, review and decide.

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

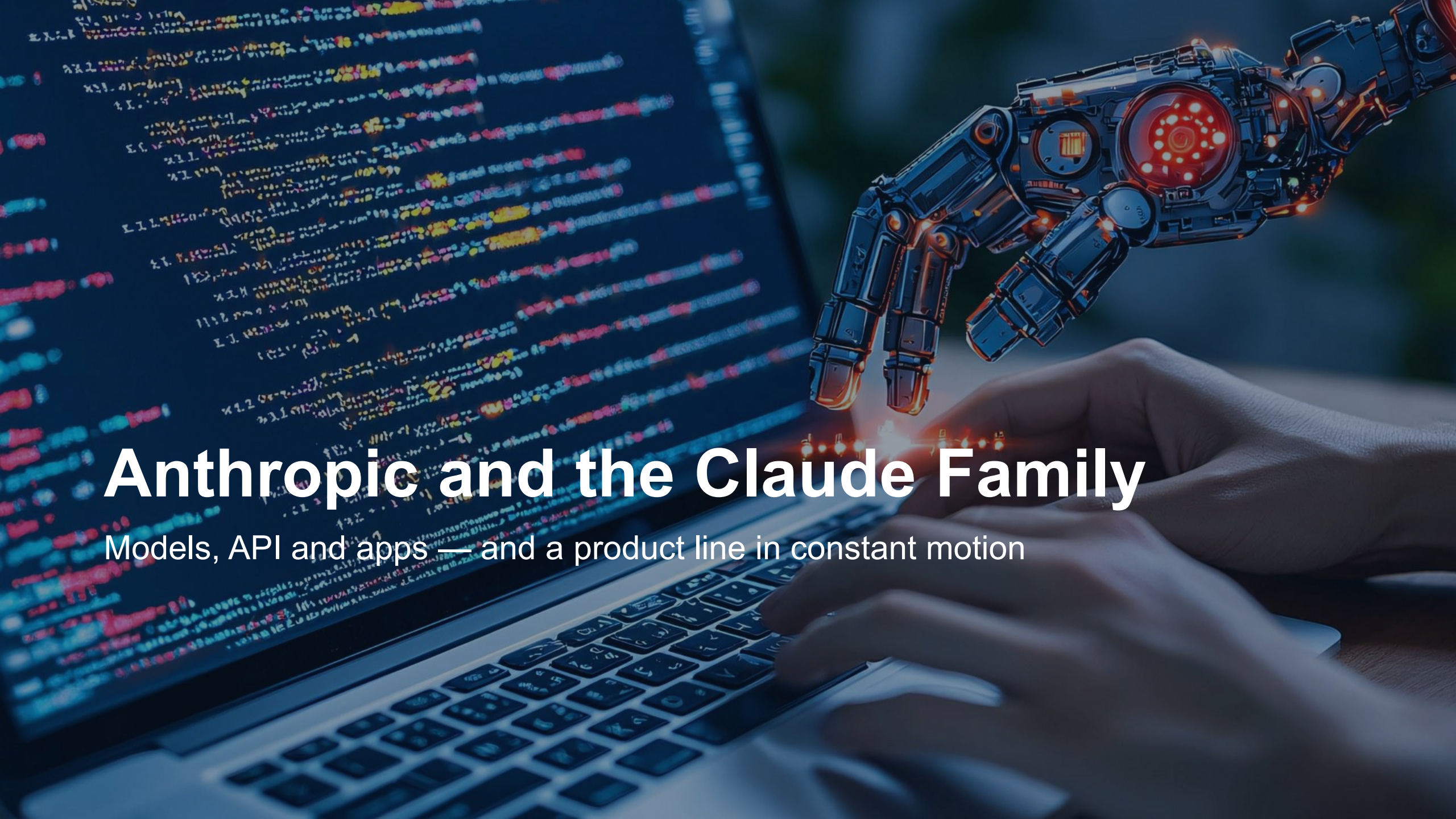
January / February 2026: Enter Claude



The only development tool I need

Claude Code is my daily companion — my IDE stays closed

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.



Anthropic and the Claude Family

Models, API and apps — and a product line in constant motion

Anthropic

Founded 2021

By Dario and Daniela Amodei and colleagues from OpenAI

San Francisco

Public benefit corporation

IPO reportedly planned for November 2026 — confidential filing in June

Focus on AI safety

Research and safety at the core of model development

Claude

The family of models and the products built on them

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

The Models

Haiku 4.5

Fast and lowest cost

Simple, high-volume tasks

API per MTok (in / out):
$1 / $5

Sonnet 5.5

Balanced workhorse

Good value — but skipped our rules

API per MTok (in / out):
$2 / $10

Opus 5.5

Most capable for daily work

My default model

API per MTok (in / out):
$4 / $20

Fable 5.1

Mythos-class frontier model

Additional safeguards

API per MTok (in / out):
$10 / $50

Claude Mythos itself is not publicly available (Project Glasswing). Versions and prices: October 2026 — they change fast.

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Life Is Too Short for Bad AI



The tools are similar — the model is not

- Claude Code, Codex, Copilot, Windsurf, Devin: the differences are small
- What counts is a strong model
- “AI isn’t working” often means: cheap AI isn’t working



Our Sonnet experiment

- Two days with Sonnet instead of Opus
- It overlooked our rules — “I just overlooked it”
- Had that been my only experience, I’d still write every line myself
- Opus is the default — Fable for complex designs and code analysis
- Benchmarks change weekly

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

The Apps

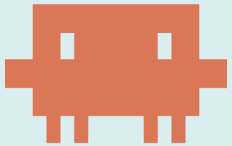
Claude
Chat on web, desktop and mobile — now including everything Cowork did

Claude Docs (beta)
Documents drafted section by section, shared and co-edited

Claude Slides (beta)
Decks from the conversation, export to PowerPoint or PDF

Claude Design
Designs, mockups and prototypes right in the chat

Claude Code
The coding agent: terminal, desktop, IDE, web — a product of its own



© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.



Claude Code

The coding agent — faces, plans and prices

The Three Faces of Claude Code

CLI

- `claude` in any terminal
- The reference implementation
- Scriptable and headless, works over SSH and in CI

Desktop App

- Several sessions side by side
- Diff review
- Local, remote and cloud sessions

IDE Extensions

- VS Code and forks, JetBrains
- Inline diffs
- Current file and selection as context

Same engine underneath: same `CLAUDE.md`, settings, skills, MCP servers and subscription — switch freely

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Plans and Pricing

Individual plans — Claude Code included from Pro upward

Free

$0

Chat, web search, files
No Claude Code

Pro

$20

per month ($17 billed annually)

Max 5x

$100

per month
5× the usage of Pro

Max 20x

$200

per month
20× the usage of Pro
My plan

Team and Enterprise — 2 to 150 seats on Team

Team — Standard seat
$25 / month — $20 annually

Team — Premium seat
$125 / month — 5× the usage

Enterprise
$20 / seat + usage at API rates, or custom

USD, excluding tax, as of October 2026 — claude.com/pricing

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

$200 a Month? Seriously?

$10

per working day ($200 ÷ 20 days)

→

10×

developer productivity — or more

The most profitable investment in developer productivity I can imagine

No faster laptop, no better chair comes close. Codex & Co. cost about the same — it's one market.

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

How I Manage My Budget

Weekly limit resets Saturday, 2 PM

In heavy weeks I add about $50 of extra usage

Friday leftovers go into reviews

Thorough code reviews of whole packages — why donate usage?

80 % used by Tuesday?

Then something is wrong — check what burns the tokens

Regular screen work never hits the limit

Big refactorings and parallel agents do

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

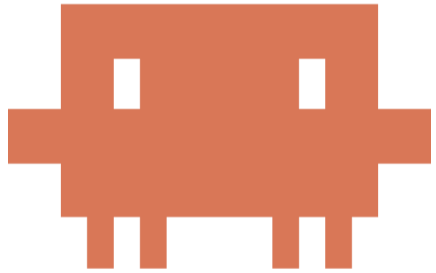
What Makes Claude Code a Coding Agent

The model — in the cloud
Opus thinks; the code stays on my disk

Runs locally on my source
Reads, searches and edits the repository

Shell and tools
Compile, unit tests, profiler, git, scripts

Rules
`CLAUDE.md` and rule files — incl. what not to do



Claude Code

Built-in best practices
Plan, split into subtasks, test, verify, ask when unsure

Skills
Recipes for recurring tasks, loaded on demand

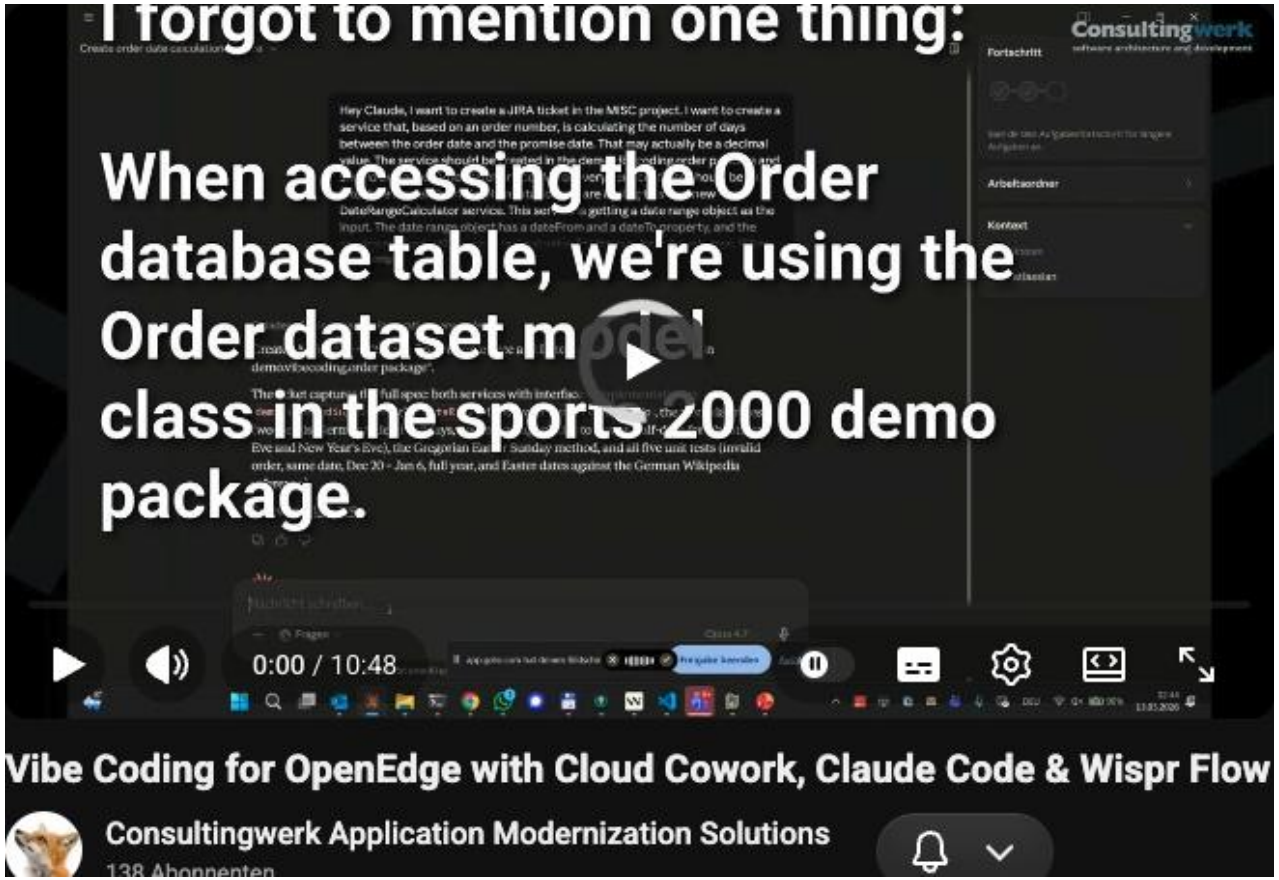
MCP integrations
Jira, OpenEdge AI Assistant, frameworks

Memory
What it learns from corrections — and where it lives

Hooks
Deterministic automation, e.g. compile after edit

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

See It in Action First



Vibe Coding for OpenEdge

Part 1: a ticket dictated with Wispr Flow — a trivial requirement grows while talking: holidays, Gregorian Easter, test requirements

Part 2: Claude Code redesigns an AppBuilder program — and keeps the AppBuilder markup intact

▶ youtube.com/watch?v=0HYVUrXoRao

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.



Teaching Claude Our Code

Configuration, context and rules

Teaching Claude Takes Work

Not install-and-go: it took us months — and it never stops

Coding and workflow standards

How we write, name, format and commit

How to compile

`build-modified` first — full build only when needed

What to do with app servers

Trim PASOE agents before a schema load

And what not to do

PASOE won't start because a chat tool blocks the port? Never move PASOE to another port. Read the log — then stop and report.

Read the transcript

Trying something new, or the result isn't what we wanted? We read the detailed session transcript — where can the flow improve, and why did Claude decide wrong?

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

My Setup: 8 Playgrounds for Claude Code

Laptop

OE 12.2 checkout	**OE 12.8** checkout
OE 13.0 checkout	**OE 13.1** checkout

PC

OE 12.2 checkout	**OE 12.8** checkout
OE 13.0 checkout	**OE 13.1** checkout

= 8 ready-to-go playgrounds

one Git repository of the SmartComponent Library, four checkouts per machine

Keep it simple

- One environment per OpenEdge release we develop, maintain and test: 12.2, 12.8, 13.0, 13.1
- Parallel Claude sessions without stepping on each other
- Framework database prepared per branch — its content versioned with the code
- PASOE instances are not rebuilt per branch
- **No separate agent harness or agent management software required**

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Our Setup in Numbers

3

`CLAUDE.md` files (repo root, ABL, UnitTests)

6

rule files imported via @

32

skills + 9 slash commands

3

MCP servers

139

knowledge entries in 8 topics

0

auto memory — switched off

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Project Context the Agent Needs

Existing source code

- The legacy application
- The modernization framework — a reference implementation of modern ABL

OpenEdge AI Assistant

- ABL language reference
- Progress Knowledge Base

Database schema

- Our MCP server provides the schema
- Alternative: `CLAUDE.md` says where the .df file lives

Without context the agent guesses — with context it follows your patterns

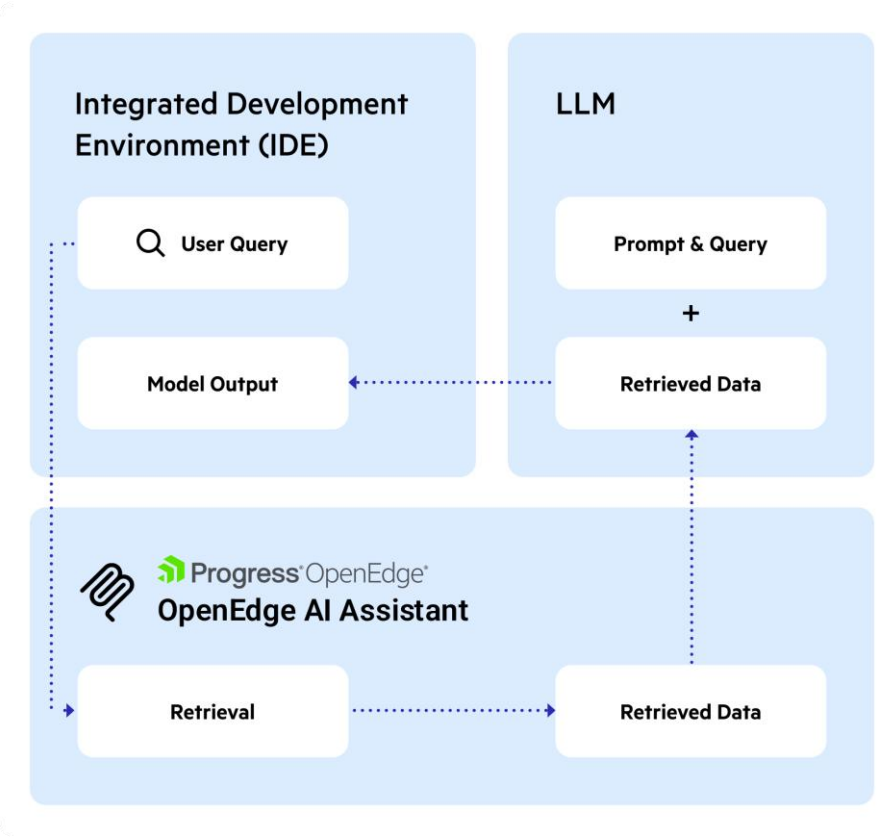
© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

What Is the OpenEdge AI Assistant?

The AI Assistant helps you to seamlessly integrate into Visual Studio Code (VS Code), Devin and other IDEs of your choice, which allows you to:

- Automate repetitive tasks like boilerplate code, refactoring and unit test generation
- Leverage your organization's rules and documentation at query time for compliant, aligned outputs
- Modernize outdated ABL into OpenEdge 12-ready syntax and practices
- Get real-time, in-context support that adapts as your project evolves

The OpenEdge AI Assistant is proudly hosted on Progress Data Cloud.



Source: progress.com/openedge/openedge-ai-assistant

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Getting the OpenEdge AI Assistant

VS Code extension

"Progress OpenEdge AI Assistant" in the Marketplace — also serves Claude Code, Devin and others

Python scripts

For command-line coding agents: a login script and `mcp-proxy.py`, published by Progress

Prerequisites

- OpenEdge 12.2, 12.8 or 13.x, active maintenance
- Python 3.9 or higher
- API key from Progress Data Cloud

New: no CVP membership required

Progress dropped the CVP requirement last week — every customer under maintenance can use it

Setup in detail: Progress has its own sessions on the OpenEdge AI Assistant at this conference

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Telling Claude When to Use It: CLAUDE.md

ABL/CLAUDE.md (excerpt)

```
## OpenEdge AI Assistant (openedge-abl-mcp-server)
### OpenEdge ABL Syntax
Use the OpenEdge AI Assistant and its 12.8 collection for ABL syntax documentation.
### OpenEdge Knowledge Base
Use its KnowledgeBase collection when troubleshooting errors, diagnosing PASOE,
AppServer or database startup issues, known bugs, workarounds and how-to questions.
Always query the KnowledgeBase collection before answering any question about
error numbers, error messages, or operational problems.
```



Lesson learned: the KnowledgeBase lags months behind — for “which update fixes X” also read the release notes

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Pre-Digested Context Beats Searching

Describe once, keep current

- Faster and more token-efficient than letting Claude search every time
- Have Claude describe key aspects of the application in Markdown files
- Instruct Claude to keep these descriptions current — in the same change

Example: our Util Helper Index

```
ABL/.devin/helpers/
  INDEX.md              one line per
                        helper class
  helpers-data.md       one line per
  helpers-gui.md        method: signature,
  helpers-json-xml.md   return type,
  ...                   purpose

Lookup rule: read INDEX.md before
grepping helper source.
Maintenance rule: update the entry
whenever a public method changes.
```

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Sample Prompt: Condensing a DF File

```
Please create a condensed version of the database schema documentation
based on SmartDB.df in this repository. It should contain:
  * table names and descriptions
  * field names and descriptions
  * indexes with fields and flags (word-index, descending, primary and unique)
Please reference the path of the DF file in the MD file and place the MD file
beside the DF file. Purpose of the MD file is to provide a coding agent
optimized access to the database schema definition.
```



Then tell `CLAUDE.md` where the MD file is — and to update it with every schema change

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.



Closing the Loop

Compile, debug, test — without an IDE

Even with Rules, Claude Makes Mistakes

Full-blown IDE

- Syntax check while you type
- Compiler errors immediately
- You know before you commit

Notepad

- Even an experienced developer writes code that might work
- You find out in production

Without a compiler, Claude is a developer with Notepad

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Teach Claude to Compile — Two Approaches



Riverside VS Code ABL extension

- The language server can be plugged in as an MCP server
- It then compiles for the agent
- Nice — but it drags an IDE back into the picture
- Not what we use



Scripts — our way

- Batch wrappers around Ant + PCT, named in `CLAUDE.md`
- `build-modified` · `build-all` · `clean`
- Single-file syntax check
- Debug listing
- Unit test runner

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

build-modified: the Rule in CLAUDE.md

```
../Scripts/sclant.cmd -f build-dev.xml build-modified   # compile modified files (try this first)
../Scripts/sclant.cmd -f build-dev.xml build-all        # compile all files
../Scripts/sclant.cmd -f build-dev.xml clean            # wipe all r-code before a full rebuild
```



Workflow rules

- “Always run a developer build after generating or modifying any ABL source file.”
- “Do not consider the task complete until the build succeeds.”
- “Attempt to fix any compile errors autonomously.”



How concrete rules must be

- “It is not on PATH … Do not search for it, do not call it from PowerShell, do not use a fully qualified C:\ path.”
- Clean + build-all after changing base classes

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Hard-Won Lessons: Compiling in Parallel

What happens

- `build-modified` compiles on 6 threads — without dependency order
- A class that uses your change may be compiled against the old r-code of the class you just changed

Typical messages — about a member or dataset you just added

```
Could not locate element 'X'
  in class ... (12927)

Dataset dsX ... does not match
  ... (13808)
```

What the rules tell Claude

- **Re-run once before diagnosing** (twice after a large pull) — only errors that survive the retry are real
- **ABL first, then UnitTests — never both at once:** UnitTests compiles its own copies of ABL classes; stale r-code shows up there as 12927
- **Full output into a file**, then search for “Error compiling” — never pipe a build through `tail`, the compile errors get lost

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Hard-Won Lessons: Git Restores Fool build-modified

What happens

- build-modified decides itself what to recompile
- After git stash pop or git checkout <ref> -- <file> it reports “0 file(s) compiled” and keeps the stale .r
- Even touching the source does not help

Why it matters

- To prove a new unit test fails without the fix, Claude temporarily reverts the fix
- The test then silently runs against the fixed r-code — and passes
- A false green

What the rules tell Claude

- Delete exactly that one .r and re-run build-modified — it recompiles as “[No r-code]”
- Otherwise never delete .r files before build-modified
- clean + build-all only for changed base classes or interfaces

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Running Unit Tests

UnitTests/CLAUDE.md

```
../Scripts/sclant.cmd UnitTests -f build-dev.xml -Dtestclass=<test class name>
```



What Claude does

- Parses results.xml (JUnit) and shows a table: class, method, status, duration, message
- “When unit tests are failing, attempt to fix the code under test (or the test itself, if the test is wrong) … then re-run”



Gotchas that cost us time

- The Ant exit code is 0 even when tests fail — the verdict is only in results.xml
- The runner executes existing r-code: run build-modified first

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Debug Listing: Finding the Real Line Number

Why

- Compile messages point to the source — runtime stack traces, protrace files and logs refer to the debug listing
- Includes and preprocessor shift lines by dozens or hundreds
- “The debug listing is the source of truth”

One file, DEBUG-LISTING, into %TEMP%

```
../Scripts/build-debug-list.cmd
  "C:\...\ABL\Consultingwerk\
    ...\SomeClass.cls"
```

Then

- Claude reads the reported line in the listing
- and deletes the temporary listing afterwards

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

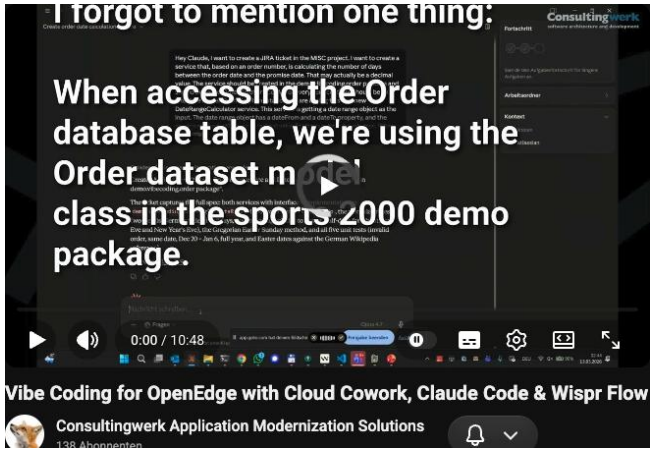
The Loop in Action

How we work in an IDE

Code → Compile → Read error → Fix

How Claude Code works

Edit → Compile → Read error → Fix → Run tests → Green → Remember



- In the video: Claude declares a variable `iF` — it collides with the ABL keyword IF; the compile fails, Claude reads the error and fixes it
- **The difference:** tell Claude to remember it — and it remembers a lot more than any of us

▶ youtube.com/watch?v=0HYVUrXoRao

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.



Rules, Memory and Workflow

What we teach Claude — and where it lives

CLAUDE.md — a Structured Overview

Loaded hierarchically

```
<repository root>
├── CLAUDE.md
│   workflow, Jira, git,
│   checkouts, memory rule
├── ABL/
│   └── CLAUDE.md
│       build and tooling
└── UnitTests/
    └── CLAUDE.md
        tests
```



Environment
Four checkouts, one per OpenEdge release · databases · MCP servers on PASOE



Conventions
“BE” = Business Entity — never abbreviate



Atlassian
Writing to Jira and Confluence · components · always link tickets



Workflow
Feature branches · commits · bug analysis · Jenkins

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

The Rule Files

```
ABL/.devin/rules/
  abl-rules.md          351 lines
  code-formatting.md    232 lines
  framework-rules.md    801 lines
  lint-rules.md         428 lines
  ui-layout.md          145 lines
  debugging.md          181 lines

Imported in ABL/CLAUDE.md:
  @.devin/rules/abl-rules.md
  ...
```



Example from debugging.md

"A 'file does not exist' error may really be the Windows 260 character path limit"



Why .devin?

A leftover from our Windsurf / Devin days — we kept it when we moved to Claude Code. It doesn't hurt.

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

From Our Rule Files: code-formatting.md

Keywords and annotations

- ABL keywords should be in lowercase.
- Add @lowercase. at the beginning of new .cls, .p, .w files.
- Do not add @lowercase. to include files.

Comments

- Comments use US English spelling and ASCII only (no em dashes, no arrows).
- New methods, properties and events get Javadoc-style comment blocks:

```
/**
 * Purpose: Description of the method
 * @param pcParam Description
 */
```

Indentation

get. / set. start in the same column as the define property statement:

```
    define public property DateFrom
        as date no-undo
    get.
    set.
```

Incorrect (do not do this):

```
    define public property DateFrom
        as date no-undo
        get.
        set.
```

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Don’t Edit These Files by Hand



Let Claude write its own instructions

- `CLAUDE.md`, rule files and skills are written by Claude — on instruction
- Structured and optimized for Claude, not for humans
- Claude knows better than I do whether something belongs in `CLAUDE.md`, a rule, a skill or an MCP tool



Like a teacher with a student

- A prompt does it: “Add a rule to …” or “Update the rule files with …”
- After a task: “Explain why it failed at first — and remember it in the rule files”
- Regularly an hour or two with Claude: too many rules, too few? Contradictions get merged
- Claude also advises on token efficiency

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Always-On vs. On-Demand

Always-on rules

- `CLAUDE.md` files and rule files
- Cost context in every session
- Short, general rules only

On-demand knowledge

- `.devin/knowledge/` with one INDEX.md
- Costs nothing until its index line matches the task
- Gotchas, recipes, background, design decisions

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Skill Example: GUI for .NET Dialogs

ABL/.claude/skills/create-gui-net-dialog/SKILL.md

```
---
name: create-gui-net-dialog
description: Decides how to prompt the user in
  GUI for .NET (ComplexInput versus a dedicated
  dialog class) and lays it out in the standard
  dialog design ... Use whenever the user asks
  to "prompt the user for ...", "add a dialog",
  ... also when the user does not say "dialog"
  but describes a modal prompt.
---
## Step 1 - Prefer InputPromptHelper:ComplexInput
## Step 2 - A dedicated dialog class
### Naming
### Layout: white content on top, buttons below
### Buttons, DialogResult and validation
### API of the dialog class and disposal
## Reference dialogs
## Workflow
```

What the skill teaches

- **Decide first:** standard input fields? Use `InputPromptHelper:ComplexInput` — no new class
- **Otherwise our house design:** white content panel on top, OK / Cancel on grey below, 75 × 23, 6 / 12 px gaps
- AcceptButton / CancelButton with DialogResult, all texts in the .resx, named <UseCase>DialogForm
- Reference dialogs to copy — and one not to copy

The description decides when Claude loads it — 230 lines that cost nothing until someone needs a dialog

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

More Skills from Our Repository

generate-db-schema-md

- Our DF-condensing prompt, grown into a skill with a Python script
- Verifies table, field and index counts against the DF file
- Commit the MD with the DF change — never edit it by hand

init-branch

- Creates a SmartDB for the current feature branch — or re-activates it
- Proposed when the Jira ticket implies schema or repository data changes
- Then build-modified against the new database

With explicit approval

- “ALWAYS ask the user for confirmation before creating the branch database – never run init-branch.cmd unprompted.”
- Yes / No prompt — No means stop
- Clients still connected? Show them and ask — never kill processes

35 skills in our repository today — all written by Claude, on instruction

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

The Trouble with Auto Memory

307

memory files

8

folders that had never seen each other

4 × 2

checkouts per OpenEdge release × machines



Auto memory lives in ~/.claude/projects/<path>/memory/

- One folder per checkout and machine — invisible to colleagues
- Duplicates and contradictions; lessons from 13.0 unknown in 12.8
- Who owns it? Not the team — and it doesn't move to another coding agent

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Team Memory in the Repository

```
<repository root>
├── CLAUDE.md              always loaded
├── .claude/
│   ├── settings.json      autoMemoryEnabled: false
│   └── skills/remember/
├── .devin/knowledge/      on-demand team memory
│   ├── INDEX.md           one line per file
│   └── abl/ build/ framework/ gui/ ...
├── ABL/
│   ├── CLAUDE.md
│   └── .devin/rules/*.md
└── UnitTests/CLAUDE.md
```

.claude/settings.json

```
{
  "autoMemoryEnabled": false,
  "permissions": {
    "allow": [ "Read(**)" ] }
}
```



Versioned, shared, portable

- New knowledge goes through the `remember` skill
- Testable in a feature branch, same for the whole team, readable by any agent

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Knowledge Example: Claude Got It Wrong



What went wrong (SCL-4327)

- Claude added a new class `RecordNotFoundException` to Consultingwerk.Exceptions
- A WebHandler imports two packages via wildcard `using` — both now had that short name
- ABL silently picked the new class: compiled clean, a unit test broke

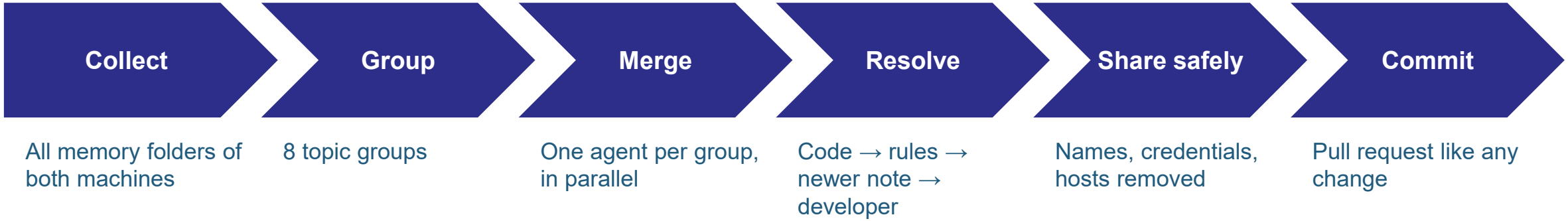
Found once, recorded via the remember skill — not repeated

.devin/knowledge/abl/wildcard-using-short-name-collision.md

```
---
name: wildcard-using-short-name-
      collision
description: Adding a class whose short
  name already exists in another wildcard-
  imported package silently changes which
  type references resolve to
tickets: SCL-4327
---
How to apply:
- Preferred: no second class with an
  existing short name
- Unavoidable: qualify the reference
- Reproduce: delete the consumer .r
```

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

How Claude Reorganized Its Own Memory



307 → 135

knowledge files — with Opus, not Sonnet: deduplicating needs judgment

Facts that differ per release

are not conflicts — they are kept and tagged with the OpenEdge release

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Maintaining the Memory

Record via the remember skill

It decides: rule file, CLAUDE.md or knowledge file

Correct instead of adding

Fix or delete wrong entries — the code decides

Memory changes are commits

Reviewed in pull requests like code

Treat Claude like a junior

When he makes a mistake, let him fix it — then tell him to remember it

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Prompting into the Ticket



Dialogue	Requirements	Design	Test requirements	Let it rest
Claude Code (or Claude chat) drafts the ticket with me — Claude asks questions	What the user needs	How we build it	How we prove it works	Days if needed — until business analyst, architect and developer are happy

```
Then: "Implement ticket ABC-1234"
```

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Sample: SCL-1581 Object Instance Customization

12 years
in the backlog

2 lines
in the ticket

~3 pages
dictated requirements

8–10
questions from Claude

19
subtasks

~8 h
implementation

The story

- “Implement Progress Dynamics-like customization in the repository of the SmartComponent Library” — a nice-to-have, but complex: 2–3 months, so never prioritized
- The backlog dried out — I dictated requirements and design constraints, Claude read the existing code and asked its questions
- Claude updated the ticket and created 19 subtasks; after my review: “implement them one after another”

The result

- Two minor bugs, caught quickly in testing
- Used internally for two weeks
- Ready to be released to customers

▶ SCL-1581 in Jira

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

From Ticket to Branch

One branch per ticket

- `feature/<ticket>` — the ticket number only, never the ticket title
- Sub-task: the parent branch is fine

Resolved automatically

- On develop Claude checks out the local branch, tracks origin or creates it from develop
- And says which

Jira follows

- Ticket moves to “In Progress” without asking
- For every ticket the work touches
- The PR moves it to “In Review” — with a description that combines a release notes draft and the design decisions

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

“PR” Means the Whole Sequence



Details

- Commit message prefixed with the ticket key
- `jira-pr-comment` skill: PR comment plus transition
- Clean up: unit test logs removed, local feature branch deleted after the merge, Claude session archived

Safety net

- Follow-ups after a merge: back into the feature branch — never commit on develop
- Worst case: throw the branch away and start again

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Guardrails That Saved Us

Analysis first

Fix only on instruction — a diagnosis request is not a fix request

Enhancement requests are never implemented

SCLER tickets move to SCL first

Released means untouchable

Closed feature ticket → new bug ticket, new branch

Shared checkouts

Never git stash; re-check the branch before every commit

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Unit Tests Are Part of “Done”

31,000

~~29,000~~

unit tests

74.5 %

~~40 → 60 %~~

coverage within a year

4

~~3~~

OpenEdge releases in the Jenkins chain

Claude is not lazy in writing tests

- Relevant unit tests run locally — automatic regression check
- Profiling harness: performance before and after a change
- All unit tests run on Jenkins on every commit
- Unit test status visible in GitHub

Jenkins chain

- 12.8 → 13.0 → 13.1 (→ 12.2 on develop)
- Each release only after the previous one is green
- SonarQube and AI reviews on top
- Claude can read code coverage from SonarQube

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

A Session Is the Short-Term Memory of One Task

Inside a session

- The full dialog with me
- Every decision and every dead end
- All tool output
- Once the task is done, most of it is irrelevant
- Relevant design decisions: fold them back into the ticket or other documentation — persisted for human and machine

What survives

Only what was written down:

- Code and commits
- `CLAUDE.md` and rule files
- Knowledge / memory files
- The Jira ticket

Everything else: assume it is gone

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Why Not One Session for Life?

Context fills up → Agent compacts → Details get lost → Tokens burn

How I work with sessions

- One session per task or ticket — `/clear` before the next one
- Before closing: write down what matters (ticket comment, rule, `remember` skill)
- `--continue` / `/resume` only to pick up the same task
- Compact deliberately at a good breakpoint (`/compact` with a focus hint)
- Start the next session from the Jira ticket

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Key Takeaways

The IDE is optional

The agent compiles, tests and commits

The model matters

Life is too short for bad AI

Teaching takes months

And pays back ten times

Rules belong in the repository

Versioned, shared, portable

Close the loop

Compile, test and fix without you

Prompt into the ticket

Then: implement ticket ABC-1234

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

My Sessions at PUG Challenge Amsterdam

Session	Day	Time	Room
Say Bye-Bye to the IDE. Here Comes Claude Code.	**Wed 4**	**16:30 – 17:30**	**New York 3**
AI-Powered OpenEdge Modernization: Faster, Smarter, Scalable	Thu 5	11:45 – 12:45	Athens
REST API Versioning: Concepts and Samples	Thu 5	16:30 – 17:30	New York 3
Cool Things Your Coding Assistant Can Do For You	Fri 6	13:45 – 14:45	New York 1

Highlighted: this session

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.



Subscribe Now to Consultingwerk's Official Newsletter!

www.consultingwerk.com/resources/newsletter

or marketing@consultingwerk.com

© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.

Questions



© 2026 Consultingwerk Application Modernization Solutions Ltd. All rights reserved.


Consultingwerk
software architecture and development